Архитектурные подходы и методы балансировки нагрузки высоконагруженных сетей в центрах обработки данных

Почебут А.С.

Научный руководитель: Лихачевский Д.В., к.т.н., доцент, декан факультета компьютерного проектирования

магистрант

Белорусский государственный университет информатики и радиоэлектроники

г. Минск

a.pochebut03@gmail.com

В работе рассмотрены современные архитектурные подходы и методы балансировки нагрузки в высоконагруженных сетях в центрах обработки данных. Анализируются принципы построения масштабируемых и отказоустойчивых архитектур, обеспечивающих равномерное распределение трафика. Приведена классификация алгоритмов балансировки, их сравнительная оценка и влияние на производительность сетевой инфраструктуры. Особое внимание уделено

применению динамических и комбинированных методов, повышающих устойчивость и эффективность работы сетей при высоких нагрузках.

ВЫСОКОНАГРУЖЕННЫЕ СЕТИ, БАЛАНСИРОВКА НАГРУЗКИ, МАСШТАБИРУЕМОСТЬ, ОТКАЗОУСТОЙЧИВОСТЬ, АРХИТЕКТУРА СЕТИ, ЦЕНТР ОБРАБОТКИ ДАННЫХ, ДИНАМИЧЕСКИЕ АЛГОРИТМЫ, МАРШРУТИЗАЦИЯ

Современные центры обработки данных (ЦОД) представляют собой сложные вычислительные комплексы, обеспечивающие функционирование облачных сервисов, систем хранения данных и корпоративных приложений. Рост объёма обрабатываемой информации и усложнение сетевых топологий создают повышенные требования к масштабируемости и отказоустойчивости инфраструктуры. В этих условиях ключевое значение приобретает эффективная балансировка нагрузки между сетевыми узлами, позволяющая обеспечить равномерное распределение трафика и стабильную работу сервисов даже при пиковых нагрузках [1].

Архитектура сетей ЦОД определяет эффективность распределения трафика, способность поддерживать высокую плотность соединений между вычислительными узлами и возможность масштабирования при увеличении числа серверов и каналов связи. Традиционно применяемая в корпоративных сетях трёхуровневая модель (Access-Aggregation-Core) обладает ограничениями по масштабируемости и склонна к образованию узких мест при росте количества соединений. Для преодоления этих ограничений в последние годы активно используются архитектуры Fat Tree и Spine-Leaf [2].

Архитектура Spine-Leaf представляет собой двухуровневую сетевую топологию, включающую уровень «позвоночника» (Spine) и уровень «листьев» (Leaf). Уровень Leaf состоит из коммутаторов доступа, к которым подключаются серверы, системы хранения данных и сетевые шлюзы. Уровень Spine образован высокопроизводительными коммутаторами, объединяющими все Leaf-коммутаторы между собой. Каждый Leaf-коммутатор соединён со всеми Spine-коммутаторами, что формирует полносвязную схему между уровнями, обеспечивает равномерное распределение трафика и минимизирует вероятность перегрузки отдельных каналов. Такая топология обеспечивает одинаковое количество переходов между любыми двумя узлами сети (обычно не более двух), что приводит к стабильной задержке передачи и предсказуемому времени отклика.

Основными преимуществами архитектуры Spine-Leaf являются масштабируемость, высокая пропускная способность и устойчивость к отказам. Благодаря особенностям топологии добавление новых серверов или стоек требует лишь подключения дополнительных Leaf-коммутаторов без перестройки всей инфраструктуры, а равномерное распределение соединений между Spine-коммутаторами позволяет эффективно

использовать суммарную полосу пропускания. Кроме того, при выходе из строя одного канала или узла трафик автоматически перенаправляется по альтернативным маршрутам через другие Spine-коммутаторы. Плюсом является и простота интеграции с программно-определяемыми сетями (SDN), так как архитектура естественно сочетается с централизованным управлением и динамическими политиками маршрутизации. К недостаткам можно отнести значительное количество межсоединений при большом числе узлов, что требует повышенной плотности портов у коммутаторов и может увеличивать стоимость оборудования.

Архитектура Fat Tree является обобщением классической структуры дерева с множественными связями между уровнями, что позволяет повысить пропускную способность и отказоустойчивость. Такая конфигурация напоминает «утолщённое дерево» (fat tree), где узлы верхних уровней имеют более широкие каналы связи (или большее количество соединений), чем узлы нижних, что позволяет каждой ветви передавать больший объём данных по мере приближения к корню дерева. Исследования показывают, что при высокой плотности связей вероятность перегрузки каналов в архитектуре Fat Tree на 15-20% ниже, чем в классической трёхуровневой модели.

В типичной реализации Fat Tree сеть состоит из трёх уровней:

- уровень доступа (Edge), который подключает серверы к нижним коммутаторам;

- агрегационный уровень (Aggregation), который объединяет несколько коммутаторов доступа и выполняет первичную маршрутизацию;

- корневой уровень (Core), который обеспечивает соединение между всеми подсетями ЦОД.

Основная идея Fat Tree заключается в многопутевой маршрутизации: между любой парой конечных узлов существует несколько равнозначных маршрутов, что позволяет использовать протоколы ECMP (Equal-Cost Multi-Path) и распределять трафик по всем доступным путям [3]. Это обеспечивает равномерную загрузку каналов и снижает вероятность перегрузки отдельных маршрутов.

К преимуществам архитектуры Fat Tree можно отнести линейную масштабируемость и высокую отказоустойчивость. Увеличение числа серверов достигается добавлением новых «поддеревьев» без модификации существующих узлов, что является существенным плюсом. Множественность маршрутов же обеспечивает сохранение связности даже при отказе нескольких каналов. Более того, за счёт использования коммутаторов одинаковой производительности на всех уровнях можно снизить затраты по сравнению с трёхуровневыми иерархиями. К ограничениям относятся более сложная логическая маршрутизация и повышенные требования к алгоритмам балансировки трафика. При

неравномерных потоках возможны временные перегрузки на отдельных узлах, если не применяется динамическая корректировка маршрутов.

Таким образом, Spine-Leaf оптимальна для большинства современных корпоративных и облачных центров обработки данных, где важны простота масштабирования и прогнозируемая задержка. Fat Tree целесообразна при проектировании крупномасштабных ЦОД, ориентированных на обработку больших объёмов параллельных потоков данных и высокие требования к отказоустойчивости.

Балансировка нагрузки в сетях ЦОД направлена на обеспечение оптимального распределения сетевого трафика между доступными вычислительными узлами, каналами связи и сетевыми устройствами. При этом балансировка позволяет повысить общую производительность, сократить задержки передачи данных и обеспечить стабильную работу сервисов при изменении интенсивности запросов. В условиях высоконагруженных сетей ЦОД процессы распределения нагрузки приобретают динамический характер: система должна не только эффективно перераспределять потоки при изменении состояния каналов, но и адаптироваться к изменению топологии, выходу узлов из строя и неравномерному распределению входящего трафика.

Методы балансировки можно классифицировать по уровням применения и алгоритмическим принципам [4]. Статические методы подразумевают распределение потоков по заранее заданным правилам. Примерами таких методов балансировки являются Round Robin и его улучшенная версия Weighted Round Robin. В классическом Round Robin запросы последовательно направляются на каждый сервер по кругу, что обеспечивает равномерное распределение нагрузки при одинаковой производительности узлов. Такой подход прост в реализации и не требует постоянного мониторинга состояния сети. Модификация Weighted Round Robin учитывает различия в вычислительных возможностях серверов: каждому узлу присваивается вес, определяющий долю запросов, направляемых на него. Это позволяет более эффективно использовать ресурсы в гетерогенных системах, где узлы отличаются по мощности и пропускной способности, сохраняя при этом предсказуемость и простоту механизма распределения. Эти методы просты в реализации, но не учитывают текущее состояние каналов и могут приводить к перегрузке отдельных узлов при изменении нагрузки.

Динамические методы в свою очередь принимают решения о маршрутизации на основе актуальных данных о загрузке каналов, задержках и ошибках передачи. Наиболее распространёнными алгоритмами, работающими по такому принципу, являются Least Connections, Dynamic Ratio Load Balancingи Adaptive Hash Routing. Принцип работы метода Least Connections основан на принципе выбора того сервера,

который в данный момент обслуживает наименьшее количество активных соединений. Такой подход позволяет учитывать текущую загрузку каждого узла и избегать ситуаций, когда отдельные серверы оказываются перегруженными, несмотря на наличие свободных ресурсов у других. Алгоритм особенно эффективен в системах с неравномерным распределением запросов по времени или разной продолжительностью сеансов, так как он обеспечивает более равномерное использование вычислительных мощностей и сокращает время отклика. В отличие от статических схем, динамические методы требуют постоянного мониторинга состояния сети, но обеспечивают значительно более высокую эффективность при переменной нагрузке, снижая вероятность образования «узких мест».

Существуют также и комбинированные подходы, которые сочетают преимущества статических и динамических алгоритмов. Например, распределение по весам с последующей коррекцией маршрутов на основе мониторинга трафика. Такие схемы применяются в современных балансировщиках уровня L4-L7 и позволяют адаптироваться к изменяющимся условиям сети.

Повышение масштабируемости достигается за счёт децентрализации управления и распределения функций маршрутизации между несколькими уровнями сети [5]. Использование протоколов динамической маршрутизации, таких как ECMP (Equal-Cost Multi-Path), позволяет направлять пакеты по нескольким равнозначным маршрутам, равномерно распределяя нагрузку. Важную роль играет внедрение технологий виртуализации сетевых функций (NFV) и программно-определяемых сетей (SDN), которые обеспечивают централизованное управление потоками данных и гибкую перенастройку маршрутов без физического вмешательства [6]. Это особенно актуально для центров обработки данных, где перераспределение нагрузки может происходить в реальном времени в зависимости от состояния узлов и сервисов.

Повышению отказоустойчивости способствует использование избыточных соединений и резервных путей передачи данных. В современных архитектурах это реализуется через дублирование критически важных каналов и применение механизмов автоматического переключения при обнаружении отказа оборудования или канала связи. Такие решения позволяют минимизировать время простоя сервисов и предотвратить потерю данных при сбоях.

Всё более широкое распространение получают гибридные архитектуры, объединяющие элементы традиционных сетей и облачных платформ [7]. Подобные решения позволяют перераспределять нагрузку между физическими и виртуальными сегментами сети в зависимости от текущей производительности и приоритетов сервисов. Применение

контейнеризации и микросервисной архитектуры дополнительно повышает гибкость инфраструктуры: отдельные компоненты системы могут масштабироваться независимо друг от друга, что снижает риск перегрузки и повышает общую устойчивость ЦОД.

Архитектурные подходы и методы балансировки нагрузки являются ключевыми факторами, определяющими эффективность функционирования высоконагруженных сетей в центрах обработки данных. Наиболее перспективными направлениями развития являются использование масштабируемых архитектур типа Spine–Leaf и Fat Tree, применение динамических алгоритмов маршрутизации и внедрение программно-определяемых механизмов управления. Эти решения позволяют обеспечить высокую производительность, устойчивость к сбоям и гибкость в условиях постоянно растущих требований к сетевой инфраструктуре. Таким образом, сочетание децентрализованных архитектурных принципов, виртуализации и интеллектуального управления потоками данных формирует основу для создания высокопроизводительных и отказоустойчивых сетей нового поколения.

Список литературы

1. Таненбаум Э. Компьютерные сети / Э. Таненбаум, Д. Уэзеролл. – Изд. 6-е. – СПб.: Питер, 2023. – 816 с.

2. Докучаев, В. А. Архитектура центров обработки данных : учеб. пособие / В. А. Докучаев, А. А. Кальфа, Д. В. Гадасин. – Москва: МТУСИ, 2018. – 155 с.

3. ECMP и балансировка в ЦОД (электронный ресурс) https://linkmeup.ru/blog/903/ (дата обращения 18.10.2025)

4. Vivek V. Load Balancing in Data Center / V. Vivek, A. Patil, S. Kumar. – Amsterdam: Juniper Networks, 2023.

5. Al-Fares, M. A Scalable, Commodity Data Center Network Architecture / M. Al-Fares, A. Loukissas, A. Vahdat // ACM SIGCOMM Computer Communication Review. – 2008. – Vol. 38, № 4. – P. 63-74.

6. Kreutz, D. Software-Defined Networking: A Comprehensive Survey / D. Kreutz, F. Ramos, P. Verissimo // Proceedings of the IEEE. – 2014. – Vol. 103, № 1. – P. 14-76.

7. Greenberg, A. The Cost of a Cloud: Research Problems in Data Center Networks / A. Greenberg, J. Hamilton, D. Maltz // ACM SIGCOMM Computer Communication Review. – 2009. – Vol. 39, № 1. – P. 68-73.